REMARKS/ARGUMENTS

These remarks are submitted responsive to the Final Office Action of December 9, 2005 (hereafter Office Action). As this response is timely filed within the 3-month shortened statutory period, no fee is believed due.

Claims 1-6, 11, and 16-21 were rejected in the Office Action under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent Publication No. 2003/0069908 to Anthony, *et al.* (hereinafter Anthony) in view of "Understanding UML: The Developer's Guide With a Web-Based Application in Java", by Fowler, *et al.* (hereinafter Fowler), and in further view of U.S. Patent Publication No. 2002/0016828 to Daugherty, *et al.*, (hereinafter Daugherty). Claims 7-8 and 12-15 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Anthony, in view of Fowler and in further view of "Laura Lemay's Web Workshop JavaScript", by Lemay, *et al.* (hereinafter Lemay).

Applicant has amended independent Claims 1, 7, 11, and 16 to emphasize certain aspects of Applicant's invention. The amendments are supported throughout the Specification. (See, e.g., Specification, p. 3, lines 18-23 and lines 28-29; p. 4, lines 9-10; p. 6, lines 13-14; and p. 7, lines 2-4.) No new matter has been introduced by virtue of the claim amendments.

Applicant's Invention

It may be useful at this juncture to reiterate certain aspects of Applicant's invention. One embodiment of the invention, typified by independent Claim 1, is a method for defining a markup language representation of state chart data. The method can include loading state chart data corresponding to a state chart diagram, the state chart data being loaded through an interface to a state machine modeling tool. The state chart diagram, more particularly, can specify the behavior of a plurality of objects. The corresponding state chart data can specify possible life-cycle states for each object as well as the behavior of the objects in each specified state.

The state chart data, moreover, can be specified according to a modeling language. The modeling language can be the uniform modeling language, UML, or other modeling language. (Specification, p. 7, lines 2-7.)

The method can further include generating header data in accordance with a selected markup language and retrieving a state name and state transition data from the state chart data for each state specified in the state chart data. The state transition data can specify event occurrences for transitioning from one state to another. The method then proceeds to format the retrieved state names and corresponding state transition data. In particular, the step results in the state names and corresponding state transition data being formatted according to the selected markup language. The header data along with the formatted state names and state transition data is then saved in a document formatted according to the selected markup language.

The Claims Define Over The Prior Art

As already noted, independent Claims 1, 11, and 16 were rejected as unpatentable over Anthony in view of Fowler and Daugherty. Applicant respectfully submits, however, that the combination fails to teach or suggest every feature recited in the claims.

Anthony, Fowler, and Daugherty fail, both alone and in combination, to teach or suggest the formatting of state chart data, state names, or state transition data in a selected markup language. Anthony teaches the translation of XML documents using a fist document type definition (DTD) into XML documents using a second DTD. Anthony's XML document translation is explicitly described in a language cited at pages 2 and 13 of the Office Action as follows:

"the problem of translating an XML document that has a first DTD into an XML document that has the same semantics but has a second DTD can be solved by translating from the first XML document into a representation of the semantics and then from the representation of the semantics to a second XML document. The semantic representation can be anything which is able to represent the semantics shared by the XML documents. For example, the OCF XML document 1901 can be represented by an Ariadne taxonomy model, and the translation can be done as shown in FIG. 20." (Paragraph 0366.)

In the most general sense, the term semantics pertains to meanings assigned to symbols and sets of symbols in a language. (See, e.g., www.m-w.com/dictionary/semantics.) In the context of computer languages, the term is frequently used to differentiate the meaning of an instruction from its format. (See, e.g., <www.webopedia.com/ TERM/s/semantics .html>.) A semantic representation of an XML document, as described in Anthony, however, provides none of the nuances of state chart data corresponding to a state chart diagram, as recited in each of the independent claims. Anthony's semantics do not, for example, specify life-cycle states for an object or the behavior exhibited by an object in a specific state, as further recited in the independent claims.

At pages 3 and 13 of the Office Action, it is acknowledged that Anthony fails to explicitly disclose aspects relating to state chart data. It is asserted, however, that Fowler discloses each of the recited features by teaching the use of a Unified Modeling Language (UML) to generate a state diagram indicating the behavior of two objects or items in an order processing system. But Fowler merely discloses what is already widely known, namely, that a UML state diagram can be used to describe the behavior of a system.

Anthony's XML-to-XML translations, however, have nothing to do with Fowler's general description of a particular application of a UML, and Applicant respectfully reiterates that state diagrams and XML documents are concepts entirely distinct from one another. More fundamentally, even when combined, Anthony and Fowler do not teach or suggest transforming state chart data specified according to a modeling language into a markup language representation, as recited in amended independent Claims 1, 11, and 16.

Anthony's translation is merely from one XML format to another. Moreover, Anthony's XML translation uses a specific source DTD corresponding to the source XML document (a text document) and then translates the source DTD to a target DTD in order create a target XML document (another text document). (Paragraphs 0009 and 0011; Abstract.) But a UML does not provide a DTD. Accordingly, one of ordinary skill, looking at the UML state-machine diagram in Fowler and the XML-to-XML translation of Anthony, is left with no teaching as to how to use Anthony to transform the UML state-machine diagram of Fowler – which lacks the DTD that Anthony depends on for XML-to-XML translations – into an XML-formatted document.

As described in the above-quoted portion, Anthony discloses that a first XML document can be translated into a second by converting the former document into a representation of semantics and then from the representation into the second XML document. This approach, however, requires that the semantic representation in fact represent "semantics shared by" both XML documents. More fundamentally, however, a semantic representation of an XML documents neither teaches nor suggests a mechanism for translating a UML state-machine diagram, as in Fowler, into XML documents that are the exclusive focus of Anthony.

Accordingly, Anthony and Fowler, alone and in combination, fail to teach or suggest any mechanism by which data specified according to a modeling language can be transformed into an XML format. Specifically, neither of the references teaches or suggests retrieving state names with corresponding state transition data from state chart data, specified according to a modeling language, and then formatting the state names and corresponding state transition data according to a selected markup language, as recited in independent Claims 1 and 16. Similarly, neither reference teaches or suggests, for example, a markup language formatter for formatting in a markup language the state chart data and component state actions specified according to a modeling language, as recited in independent Claim 11.

At page 5 of the Office Action, it is asserted that Daugherty discloses generating header data in accordance with a selected markup language. Nothing in the generation of header data according to Daugherty remotely teaches or suggests the features recited in amended independent Claims 1, 11, and 16. Specifically, Daugherty does not remotely teach or suggest transforming data specified according to a modeling language into an XML format, as recited in each of the claims.

Anthony is also cited at page 8 of the Office Action with respect to Claim 7 as teaching the formatting of state chart data according to a selected markup language. Fowler again is cited as teaching the generation of a state diagram. Lemay is additionally cited as teaching the "combining of a JavaScript program with an HTML document for performing certain functions."

Applicant respectfully reasserts the arguments already presented regard Anthony and Fowler: neither Anthony's XML-to-XML translations nor Fowler's UML state diagrams remotely suggest re-formatting state chart data, initially specified into a modeling language, into a markup language representation, as recited in Claim 7. As already pointed out, Anthony provides only the translation from one XML format into another. Whatever Fowler teaches, generally about UMLs, the combination neither teaches nor suggests any mechanism by which state chart data specified according to a modeling language can be transformed into a markup language representation.

Claim 7 expressly recites an add-in script that is used in conjunction with a state machine tool, which generates state chart data specified according to a modeling language. The add-in script formats the state chart data specified according to a modeling language into a markup language representation. Lemay describes extending an HTML document using JavaScript. The example Lemay describes is using JavaScript to create a Web page that displays a particular greeting depending on the time of day.

(pp. 8-9.) The application of JavaScript in the context of HTML documents in order to produce "a relaxed program environment," however, suggests nothing about an add-in script that transforms modeling language state chart data into an XML representation, as taught, as recited in Claim 7. In particular, the extension of HTML programming using JavaScript does not teach or suggest an add-in script that formats state chart data specified according to a modeling language into a markup language representation.

Anthony is yet again cited with respect to Claim 14, it being asserted that Anthony discloses generating a model of an XML document using a modeling system. Fowler is cited as teaching generating a UML state diagram having state names and transition data. Nonetheless, neither reference discloses an add-in script defining a markup representation of a UML-specified state chart diagram or a state-machine run-time engine executing the markup language representation. Lemay is cited as disclosing both of the latter features.

Lemay explicitly describes extending HTML documents using JavaScript. However, Lemay's description of a JavaScript does not suggest an add-in script that leads to a markup language representation of a UML-specified state chart diagram produced by a state-machine modeling tool, as recited in Claim 14. Lemay's JavaScript begins and ends with an HTML document. Accordingly, Lemay fails to teach or suggest Applicant's invention. An HTML document is not a UML-specified state chart diagram, and Lemay provides no mechanism to transform the latter into the former. With Applicant's invention, the UML-specified state chart document is transformed into a markup language representation by an add-in script. Lemay's JavaScript does not provide this capability. Even when combined, therefore, Anthony, Fowler, and Lemay fail to teach or suggest every feature of Claim 14.

For the reasons already stated, none of the references, alone or in combination, teach or suggest every feature recited in independent Claims 1, 7, 11, 14, and 16. Accordingly, Applicant respectfully maintains that independent Claims 1, 7, 11, 14, and 16 each define over the prior art. Applicant further respectfully submits that whereas the remaining claims each depend from one of the independent claims while reciting additional features, each of the dependent claims also defines over the prior art.

CONCLUSION

Applicant believes that this application is now in full condition for allowance, which action is respectfully requested. The Applicant requests that the Examiner call the undersigned if clarification is needed on any matter within this Amendment, or if the Examiner believes a telephone interview would expedite the prosecution of the subject application to completion.

Respectfully submitted,

Date: March 9, 2006

[illegible]

Gregory A. Nelson, Registration No. 30,577
Richard A. Hinson, Registration No. 47,652
Marc A. Boillot, Registration No. 56,164
AKERMAN SENTERFITT
Customer No. 40987
Post Office Box 3188
West Palm Beach, FL 33402-3188
Telephone: (561) 653-5000